How can I identify cases used by an estimation command using e(sample)?

Authored by

stats writer

July 1, 2024

The e(sample) command is a useful tool for identifying the cases used by an estimation command. By using this command, the user can easily identify the specific cases that were used in the estimation process. This can be helpful in understanding the results of the estimation and ensuring that all relevant cases were included in the analysis. The e(sample) command can also be used to subset or manipulate the data based on the cases used in the estimation, allowing for further analysis and exploration. Overall, the e(sample) command is a valuable tool for identifying and working with specific cases in an estimation command.

How can I identify cases used by an estimation command using e(sample)? | Stata FAQ

When performing data analysis, it is very common for a given model (e.g. a regression model), to not use all cases in the dataset. This can occur for a number of reasons, for example because if was used to tell Stata to perform the analysis on a subset of cases, or because some cases had missing values on some or all of the variables in the analysis. To allow you to identify the cases used in an analysis, most Stata estimation commands return a function that takes on a value of one if the case was included in the analysis, and zero otherwise (for more information see our Stata FAQ: How can I access information stored after I run a command in Stata

(returned results)?). Below we show how this can be useful in
two common situations. Many more situations exist, once you're aware of this
function and how it works, you'll recognize them. The examples below use
two different versions of the hsb2 dataset. Both versions contain information on
200 high school
students, including their scores on a series of standardized tests, and some demographic information.

When analyzing a subset of data

In this example we run a regression model predicting student's reading scores based on
their scores for math, and science. However, we use if to indicate that we want to
run our model on only those cases where the variable write is greater
than or equal to 50. Below we see the output for this regression. Note that 128
observations were used in the analysis, rather than the full 200, because we

restricted the sample using if.

use https://stats.idre.ucla.edu/stat/stata/notes/hsb2, clear

regress read math science if write>=50

Source | SS df MS Number of obs = 128
----------+---------- F(2, 125) = 43.61
Model | 4595.51237 2 2297.75618 Prob > F = 0.0000
Residual | 6585.72982 125 52.6858386 R-squared = 0.4110
----------+---------- Adj R-squared = 0.4016
Total | 11181.2422 127 88.0412771 Root MSE = 7.2585

--
read | Coef. Std. Err. t P>|t|
----------+---
math | .4952647 .0898744 5.51 0.000 .3173921 .6731373
science | .2960287 .0942091 3.14 0.002 .1095772 .4824803
_cons | 11.7755 4.86182 2.42 0.017 2.153353 21.39764
--

Once we have run our model, we can generate

predicted values using the predict command. Below generate a new variable, p1, which contains the predicted values for each case. When we use summarize to examine the predicted values, we see that predict that the variable p1 has 200 observations, but the model from which these predictions was made used only 128 observations. Predicted values were generated for both the 128 cases where write>=50 and the 72 cases where write<50 (who were not used to estimate the model). Generally, we don't want to use a model estimated on one sample (in our case, observations where write>=50) on a different sample (observations where write<50). This is particularly true in cases like this one, where we know there is a systematic difference between the samples.

predict p1
(option xb assumed; fitted values)

```
summarize p1

Variable | Obs Mean Std. Dev. Min Max
-------------+--------------------------------------------------------
p1 | 200 53.1978 6.875587 40.55244 70.23441
```

We can use e(sample) to generate predicted values only for those cases used to estimate the model. Below we use predict to generate a new variable, p2, that contains predicted values, but this time we add if e(sample)==1, which indicates that predicted values should only be created for cases used in the last model we ran. This time Stata tells us that we have generated 72 missing values. There are 72 cases where write<=50 in the dataset, rather than predicted values, these cases were given missing values for p2. Summarizing the data again

```
predict p2 if e(sample)==1
(option xb assumed; fitted values)
```

(72 missing values generated)

sum p2

Variable | Obs Mean Std. Dev. Min Max

----------+----------

p2 | 128 56.11719 6.015408 42.64159 70.23441

For model comparison

When we want to compare nested models, the models must be estimated on the same sample in order for the comparison to be valid. When a dataset contains missing values, adding additional predictor variables to a model often reduces the number of cases available for a given model. In this example we fit a model where write predicts read, and compare the fit of this model to a model that contains math and science as well as write as predictors. We will compare the two models using a likelihood ratio test (i.e. the command lrtest). Below we first run a regression model where the variable read is predicted

by the variable write and store the estimates from that model as m1 using
the command estimates store m1.

```
use  https://stats.idre.ucla.edu/stat/stata/faq/hsb2_mar, clear
regress read write

Source | SS df MS Number of obs = 170
-------------+------------------------------ F( 1, 168) = 94.38
Model | 6188.25135 1 6188.25135 Prob > F = 0.0000
Residual | 11014.7428 168 65.563945 R-squared = 0.3597
-------------+------------------------------ Adj R-squared = 0.3559
Total | 17202.9941 169 101.792865 Root MSE = 8.0972

------------------------------------------------------------------------------
read | Coef. Std. Err. t P>|t|
-------------+----------------------------------------------------------------
write | .6496086 .0668652 9.72 0.000 .5176042 .7816129
_cons | 17.65687 3.589724 4.92 0.000 10.5701 24.74365
------------------------------------------------------------------------------

estimates store m1
```

Below we run a second model where read is predicted

by write, math,
and science. We store the estimates from this model as m2.

reg read write math science

Source | SS df MS Number of obs = 141
----------+---------- F(3, 137) = 47.27
Model | 7560.153 3 2520.051 Prob > F = 0.0000
Residual | 7304.15906 137 53.3150296 R-squared = 0.5086
----------+---------- Adj R-squared = 0.4979
Total | 14864.3121 140 106.173658 Root MSE = 7.3017

--
read | Coef. Std. Err. t P>|t|
----------+---
write | .2143165 .0915771 2.34 0.021 .0332291 .3954039
math | .3973615 .1020276 3.89 0.000 .1956088 .5991141
science | .3108218 .0905435 3.43 0.001 .1317781 .4898654
_cons | 3.851736 4.091921 0.94 0.348 -4.239757 11.94323
--

estimates store m2

Now that we have estimated the two models and stored the results, we want to test whether the model that contains
write, math, and science fits significantly better
than the model that contains only write as a predictor. One way to do this is using a likelihood ratio
test, which is what is done below with the command lrtest m1 m2. However this command generates an error message.
It turns out, the models were not estimated on the same number of cases. In order for the test to be valid,
the two models must be run on the same cases, clearly this is not the case. Looking at the error message and the output from our regressions we see that the
model using only write as a predictor was run on 170 cases, while the model that contained
write, math, and
science as predictors was run on 141 cases. The only difference between these two models is the
addition of the variables
math and science, indicating that the difference in sample size for the two models is due to missing data on

math, and science.

lrtest m1 m2
observations differ: 141 vs. 170
r(498);

So how do we make sure that the two models contain the same number of cases? First,
we run the model with write, math, and science as predictors, and store the estimates
as m2. Then we use the generate command (gen) to create a new variable
called sample that is equal to the function e(sample). In other words the variable sample
is equal to one if the case was included in the last analysis (i.e. the model we just
ran) and zero otherwise.

regress read write math science

Source | SS df MS Number of obs = 141
----------+---------- F(3, 137) = 47.27
Model | 7560.153 3 2520.051 Prob > F = 0.0000
Residual | 7304.15906 137 53.3150296 R-squared =

0.5086

---+--- Adj R-squared = 0.4979

Total | 14864.3121 140 106.173658 Root MSE = 7.3017

---

read | Coef. Std. Err. t P>|t|

---+---

write | .2143165 .0915771 2.34 0.021 .0332291 .3954039

math | .3973615 .1020276 3.89 0.000 .1956088 .5991141

science | .3108218 .0905435 3.43 0.001 .1317781 .4898654

_cons | 3.851736 4.091921 0.94 0.348 -4.239757 11.94323

---

estimates store m2

generate sample = e(sample)

Now we can use the variable sample to run the model with only write as

a predictor

regress read write if sample==1

Source | SS df MS Number of obs = 141

---+--- F(1, 139) = 70.12

Model | 4984.37291 1 4984.37291 Prob > F = 0.0000
Residual | 9879.93915 139 71.0786989 R-squared = 0.3353
----------+---------- Adj R-squared = 0.3305
Total | 14864.3121 140 106.173658 Root MSE = 8.4308

--
read | Coef. Std. Err. t P>|t|
----------+---
write | .6479233 .0773728 8.37 0.000 .4949436 .800903
_cons | 17.43003 4.1396 4.21 0.000 9.245303 25.61475
--

estimates store m1

Now we can use the lrtest command again, to test whether the model with write, math and science as predictors fits significantly better than a model with just write as a predictor.

lrtest m1 m2
Likelihood-ratio test LR chi2(2) = 42.59
(Assumption: m1 nested in m2) Prob > chi2 = 0.0000